Plan de Gestión de Riesgos
Sistema de Gestión de Laboratorios UPC

Alexandra Tinjacá Cortés
Facultad de Ingeniería
Universidad Piloto de Colombia
Bogotá, Colombia
Email: alexandra-tinjaca@upc.edu.co

Juan Pablo Camacho Peñata
Facultad de Ingeniería
Universidad Piloto de Colombia
Bogotá, Colombia
Email: juan-camacho7@upc.edu.co

Miguel Ángel Ardila García
Facultad de Ingeniería
Universidad Piloto de Colombia
Bogotá, Colombia
Email: miguel-ardila@upc.edu.co

Daniel Santiago Parra Escobar
Facultad de Ingeniería
Universidad Piloto de Colombia
Bogotá, Colombia
Email: daniel-parra7@upc.edu.co

***Resumen*—Este documento presenta el plan de gestión de riesgos para el desarrollo e implementación del Sistema de Gestión de Laboratorios de la Universidad Piloto de Colombia. El plan identifica, analiza y establece estrategias de mitigación para los riesgos técnicos, operativos, de calendario y organizacionales asociados al proyecto. Se utiliza una metodología basada en la norma ISO 31000 para la evaluación cualitativa y cuantitativa de riesgos, considerando probabilidad e impacto en una matriz 5x5. El plan incluye monitoreo continuo y estrategias de respuesta para garantizar el éxito del proyecto dentro del cronograma y presupuesto establecido.**

***Index Terms*—Gestión de riesgos, planificación de proyectos, desarrollo de software, Spring Boot, AWS, metodología ágil.**

I. Introducción

La gestión de riesgos es un proceso crítico en el desarrollo de proyectos de software que permite identificar proactivamente amenazas potenciales y establecer estrategias efectivas para su mitigación [1]. Para el Sistema de Gestión de Laboratorios UPC, se ha desarrollado un plan comprehensivo que aborda los riesgos asociados al desarrollo técnico, despliegue en la nube, adopción por parte de usuarios y cumplimiento de cronogramas.

II. Metodología de Gestión de Riesgos

Se sigue la metodología establecida por la norma ISO 31000:2018 [2] que comprende las siguientes fases:

II-A. Identificación de Riesgos

Se utilizan técnicas como:

- Brainstorming con el equipo de desarrollo
- Análisis de lecciones aprendidas de proyectos similares
- Revisión de documentación técnica y plan de proyecto
- Análisis FODA (Fortalezas, Oportunidades, Debilidades, Amenazas)

II-B. Análisis Cualitativo

Evaluación de probabilidad e impacto utilizando una escala de 1 a 5:

- **Probabilidad:** 1=Muy Baja, 2=Baja, 3=Media, 4=Alta, 5=Muy Alta
- **Impacto:** 1=Insignificante, 2=Menor, 3=Moderado, 4=Severo, 5=Crítico

II-C. Matriz de Riesgos

Se utiliza una matriz 5x5 para clasificar los riesgos en cuatro categorías:

- **1** Bajo:Riesgos aceptables con monitoreo básico
- **2** Moderado:Requieren plan de mitigación específico
- **3** Alto:Necesitan acciones inmediatas de mitigación
- **4** Extremo:Pueden comprometer el éxito del proyecto

III. Identificación y Análisis de Riesgos

III-A. Riesgos Técnicos

Cuadro I
Riesgos Técnicos del Proyecto

Riesgo	Descripción	Prob.	Impacto	Nivel
Complejidad de Integración	Dificultades en la integración frontend-backend y con servicios en la nube (AWS)	4	4	3
Problemas de Rendimiento	La aplicación no maneja adecuadamente la carga esperada de 30 TPS	3	5	3
Vulnerabilidad de Seguridad	Exposición de datos sensibles de estudiantes y administradores	3	5	3
Problemas de Base de Datos	Inconsistencias en el modelo entidad-relación o pérdida de datos	2	5	2
Dependencia de Servicios Externos	Caídas o cambios en APIs de servicios en la nube (AWS)	3	4	3

III-B. Riesgos de Calendario y Recursos

Cuadro II
RIESGOS DE CALENDARIO Y RECURSOS

Riesgo	Descripción	Prob.	Impacto	Nivel
Retrasos en Desarrollo	Incumplimiento del cronograma de 11 semanas por complejidad técnica	4	4	3
Falta de Experiencia Técnica	Curva de aprendizaje con Spring Boot y AWS más pronunciada de lo esperado	3	3	2
Disponibilidad del Equipo	Conflictos con horarios académicos y otras responsabilidades	4	3	3
Problemas de Infraestructura	Configuración incorrecta de servicios en la nube o dominios	2	4	2

III-C. Riesgos Organizacionales y de Adopción

Cuadro III
RIESGOS ORGANIZACIONALES Y DE ADOPCIÓN

Riesgo	Descripción	Prob.	Impacto	Nivel
Resistencia al Cambio	Usuarios prefieren sistema manual actual en lugar de la nueva plataforma	3	4	3
Falta de Compromiso Institucional	La universidad no adopta oficialmente la plataforma después del desarrollo	2	5	2
Problemas de Capacitación	Usuarios finales no comprenden el uso de la plataforma	3	3	2
Expectativas No Alineadas	Diferencias entre lo desarrollado y lo esperado por los stakeholders	3	4	3

III-D. Matriz de Evaluación de Riesgos

Cuadro IV
MATRIZ DE EVALUACIÓN DE RIESGOS

I	P				
	1	**2**	**3**	**4**	**5**
5	M	M	A	E	E
4	M	M	A	A	E
3	B	M	M	A	A
2	B	B	M	M	M
1	B	B	B	B	B

Leyenda: B=Bajo, M=Moderado, A=Alto, E=Extremo
P(Probabilidad): 1=Muy Baja, 2=Baja, 3=Media, 4=Alta, 5=Muy Alta
I(Impacto): 1=Insignificante, 2=Menor, 3=Moderado, 4=Severo, 5=Crítico

IV. ESTRATEGIAS DE MITIGACIÓN

IV-A. Riesgos Técnicos

IV-A1. Complejidad de Integración:

- **Prevención:** Desarrollo iterativo con integración continua desde la semana 3
- **Mitigación:** Pruebas de integración automatizadas y validación temprana de endpoints
- **Contingencia:** Uso de mocks y stubs para desarrollo paralelo de frontend y backend[1]

IV-A2. Vulnerabilidades de Seguridad:

- **Prevención:** Uso de **AWS IAM** para control estricto de identidades y permisos, **Security Groups** para aislamiento de red, y **RDS Security Features** (cifrado en reposo con KMS, cifrado en tránsito TLS 1.2, subredes privadas en VPC). Integración con **Spring Security** para autenticación y autorización a nivel de aplicación.
- **Mitigación:** Validación de inputs y sanitización de datos en todos los endpoints[2]
- **Contingencia:** Plan de respuesta a incidentes apoyado en **AWS CloudWatch Logs**, snapshots automáticos de **RDS** para recuperación (*point-in-time restore*), y despliegues con rollback mediante **versionamiento de AMIs** o **Auto Scaling Groups** con políticas de reemplazo.

IV-B. Riesgos de calendario

IV-B1. Retrasos en Desarrollo:

- **Prevención:** Seguimiento semanal del cronograma y uso de metodología ágil
- **Mitigación:** Priorización de funcionalidades MVP.
- **Contingencia:** Uso de buffer de tiempo en semanas 9-10 para ajustes finales[3]

IV-B2. Falta de Experiencia Técnica:

- **Prevención:** Sesiones de capacitación en Spring Boot y pair programming
- **Mitigación:** Documentación técnica detallada y code reviews semanales
- **Contingencia:** Asesoría del profesor director en aspectos técnicos complejos

IV-C. Riesgos Organizacionales

IV-C1. Resistencia al Cambio:

- **Prevención:** Involucramiento temprano de usuarios en diseño de interfaces
- **Mitigación:** Desarrollo de manual de usuario comprehensivo y sesiones de capacitación
- **Contingencia:** Sistema híbrido temporal (digital y físico) durante transición

[1] Los mocks y stubs son componentes simulados que permiten el desarrollo independiente de frontend y backend. Los mocks imitan comportamientos para pruebas, mientras los stubs proporcionan respuestas predefinidas, facilitando la integración progresiva sin bloqueos entre equipos.

[2] **Validación de inputs**: Verificación estricta usando Spring Boot Validation (@NotNull, @Size, @Email). **Sanitización**: Limpieza de datos para prevenir inyecciones SQL/XSS. **Mitigación en AWS**: (1) Uso de parámetros y prepared statements mediante JPA/Hibernate, (2) Secrets gestionados con AWS Secrets Manager para evitar credenciales expuestas, (3) Restricción de acceso al RDS solo desde la instancia EC2 mediante reglas de Security Group, (4) Integración opcional con AWS WAF para filtrar tráfico malicioso (inyecciones, bots, patrones anómalos), (5) Auditoría continua mediante CloudTrail y GuardDuty.

[3] **Buffer temporal**: Reserva de 2 semanas (25 % del cronograma) para imprevistos. Semanas 9-10 destinadas a: (1) Corrección de bugs críticos, (2) Optimización de rendimiento, (3) Ajustes de usabilidad, (4) Pruebas de aceptación con usuarios reales, (5) Preparación de documentación final. Estrategia recomendada por PMI para gestión de riesgos en proyectos ágiles.

V. Plan de Monitoreo y Control

V-A. Indicadores Clave de Riesgo

- **KR1:** Porcentaje de completitud vs cronograma (meta: ±5 %)
- **KR2:** Número de bugs críticos por sprint (meta: ¡3)
- **KR3:** Tiempo de respuesta promedio de la aplicación (meta: ¡500ms)
- **KR4:** Tasa de adopción por usuarios piloto (meta: ¿80 %)

V-B. Revisiones Periódicas

- **Revisión Semanal:** Estado de riesgos en reuniones de equipo los sabados
- **Revisión Mensual:** Actualización de matriz de riesgos y estrategias de mitigación
- **Revisión por Hito:** Evaluación completa al final de cada fase del proyecto

V-C. Herramientas de Monitoreo

- GitHub Projects para seguimiento de tareas y dependencias
- Amazon CloudWatch para monitoreo y métricas en tiempo real de recursos y servicios AWS

VI. Plan de Respuesta a Incidentes

VI-A. Escalación de Problemas

- **Nivel 1:** Problemas menores - Resolución por desarrollador responsable en 24h
- **Nivel 2:** Problemas moderados - Revisión por líder de proyecto en 12h
- **Nivel 3:** Problemas críticos - Reunión de emergencia del equipo completo en 4h

VI-B. Procedimientos de Contingencia

- **Caída de Servicios:** Revertir a versión estable y diagnóstico en ambiente aislado
- **Pérdida de Datos:** Restauración desde backups automáticos de AWS
- **Problemas de Seguridad:** Bloqueo temporal de funcionalidades afectadas

VII. Conclusión

El plan de gestión de riesgos presentado proporciona un framework comprehensivo para identificar, analizar y mitigar los riesgos asociados al desarrollo del Sistema de Gestión de Laboratorios UPC. La implementación efectiva de este plan, combinada con el monitoreo continuo y la adaptabilidad del equipo, maximizará las probabilidades de éxito del proyecto dentro de los constraints de tiempo, presupuesto y calidad establecidos. La naturaleza iterativa del plan permite ajustes continuos basados en lecciones aprendidas durante el ciclo de vida del proyecto.

Referencias

[1] R. S. Pressman y B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 9th ed. New York, NY: McGraw-Hill Education, 2020.

[2] International Organization for Standardization, *ISO 31000:2018 Risk management — Guidelines*, 2018.

[3] Project Management Institute, *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*, 6th ed., 2017.

[4] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research,"*MIS Quarterly*, vol. 28, no. 1, pp. 75-105, Mar. 2004.

[5] B. W. Boehm, *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

[6] I. Sommerville, *Software Engineering*, 9th ed. Boston, MA: Addison-Wesley, 2011.